On Quantum Computers and Artificial Neural Networks

Florian Neukart[1], Sorin-Aurel Moraru[2]

*[1,2]Faculty of Electrical Engineering and Computer Science, University of Brasov, 500036 Brasov, Romania

*[1]florian.neukart@campus02.at; [2]smoraru@unitbv.ro

Abstract

Quantum computer science in combination with paradigms from computational neuroscience, specifically those from the field of artificial neural networks, seems to be promising for providing an outlook on a possible future of artificial intelligence. Within this elaboration, a quantum artificial neural network not only apportioning effects from quantum mechanics simulated on a von Neumann computer is proposed, but indeed for being processed on a quantum computer. Sooner or later quantum computers will replace classical von Neumann machines, which has been the motivation for this research. Although the proposed quantum artificial neural network is a classical feed forward one making use of quantum mechanical effects, it has, according to its novelty and otherness, been dedicated an own paper. Training such can only be simulated on von Neumann machines, which is pretty slow and not practically applicable (but nonetheless required for proofing the theorem), although the latter ones may be used to simulate an environment suitable for quantum computation. This is what has been realized during the SHOCID (Neukart, 2010) project for showing and proofing the advantages of quantum computers for processing artificial neural networks.

Keywords

Quantum Computer Science; Computational Neuroscience; Computational Intelligence; Data Mining; Artificial Intelligence

Introduction

The proposed quantum artificial neural network (QANN) makes use of simulated quantum effects for learning. Thus, the ANN is supervised and requires a teacher. For the introduced quantum artificial neural network, especially the simulation of the quantum mechanical theorems and paradigms of

- quantum parallelism,
- entanglement and
- interference

are of interest. Moreover, the simulation of quantum bits for processing the information goes along with the concept of quantum parallelism, as only such may be described by a wave function ψ that exists in Hilbert space. Certainly, all quantum effects the system benefits from are simulated, as quantum computers at these times are not available. However, despite all the negative effect on performance in terms of computation time this comes along with, the simulation of an environment suitable for processing information with quantum bits has an enormous advantage: the elimination of all irrelevant effects a real world physical system would subject on these bits. In the real world, a quantum computer would use particles for data representation, and in case of a quantum artificial neural network (QANN) these particles might represent neurons with all their features, like axons and dendrites (weights) or their thresholds and activation functions. A Qbit could be represented well by two-particle systems, as hydrogen-like ions are, which additionally to the atomic nucleus consist of one electron. Depending on the distance between nucleus and electron the Schrödinger equation, a partial differentiation describing the change of a physical system's quantum state over time,

$$\frac{d}{dt}|\psi(t)\rangle = -iH|\psi(t)\rangle \quad (1)$$

can be solved. The orbit of the hydrogen-like ion's electron may vary, depending on the energy levels. Physicists name this existence in different energy levels. The orbital position of the electron is then used to represent either 0 or 1, or according to Dirac's Bra-Ket notation, either $|0\rangle$ or $|1\rangle$ and the highest orbital position representing the latter one, the lowest the former one. Physically, the electron's position can be changed by subjecting it to a pulse of polarized laser light, which has the effect of adding photons into the system. For flipping a bit from 0 to 1, enough light needs to be added for moving the electron up one orbit. To flip from 1 to 0, still more photons need to enter the system, since overloading the electron will cause it to return to its ground state. However, the important thing is that if only half of the light

necessary to move an electron is added, the electron will occupy both orbits simultaneously, which is the before-mentioned linear superposition that allows multiple configurations to be computed at once (Heaton Research).

Important Quantum Effects and Qbits

The following two quantum effects as well as the definition of quantum bits are of special importance for the understanding of the introduced quantum artificial neural network.

Superposition

The linear superposition or coherence, which results in quantum parallelism, is represented by the quantum state

$$|\psi\rangle = \sum_i c_i |\phi_i\rangle \quad (2)$$

in the Hilbert space, with complex coefficients c_i and on set of states ϕ_i in space. Quantum mechanics dictates that the interaction of a system with its environment leads to the destruction of its superposition. This allows the conclusion that the construction of a quantum computer that will not be influenced by its own irrelevant physical properties is difficult. The coefficients c_i give the indication of the system being associated with the state ϕ_i when measurement happens. At the beginning it was mentioned that the sum of the coefficients must sum to unity, which is founded on the fact that a physical system must collapse to one basis state. Summing up, the linear superposition contains all possible configurations of a quantum system at once, until measurement is done, which leads to a collapse or decoherence. However, there is more to say about the superposition, as it is the most important quantum effect in terms of processing power – in particular it allows quantum parallelism: let assume a training set for a QANN has several hundreds of input data sets each consisting of several attributes, then the input register would at first grow and secondly U_f in a classical computer would have to be applied to every single input data set consecutively. Let further assume, 100 (fictive and not related to the inputs, but in numbers easier to describe) Hadamard transformations (gates) would be applied to every Qbit before the application of U_f, like

$$U_f(H^{\otimes n} \otimes 1_m)(|0\rangle_n |0\rangle_m) = \frac{1}{2^{\frac{n}{2}}} \sum_{0 \le x < 2^n} U_f(|x\rangle_n |0\rangle_m) \quad (23)$$

then the final state would contain 2^{100} or $\approx 10^{30}$ applications of U_f (Mermin, 2007). However, quantum parallelism allows the performance of an exponentially high quantity of U_f at once, or in other words, unitary time.

Entanglement

Unlike bits in a von Neumann machine, whose general state can only be one of the 2^n products of $|0\rangle$ and $|1\rangle$s, a general state of n Qbits is a superposition of these 2^n product states and cannot, in general, be expressed as a product of any set of 1-Qbit states. Individual Qbits that make up a multi-Qbit system cannot always be characterized as having individual state of their own - in contrast to the bits in a von Neumann machine - which is a nonproduct state and called entangled state.

The simplest states of entanglement between two Qbits are the Bell-states, also known as maximally entangled two-Qbit states (Bell, 1987):

$$|\phi^+\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A \otimes |0\rangle_B + |1\rangle_A \otimes |1\rangle_B) \quad (3)$$

$$|\phi^-\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A \otimes |0\rangle_B - |1\rangle_A \otimes |1\rangle_B) \quad (4)$$

$$|\psi^+\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A \otimes |1\rangle_B + |0\rangle_A \otimes |1\rangle_B) \quad (5)$$

$$|\psi^-\rangle = \frac{1}{\sqrt{2}}(|0\rangle_A \otimes |1\rangle_B - |0\rangle_A \otimes |1\rangle_B) \quad (6)$$

These states dictate that if two individuals A and B, often named Alice and Bob, each possesses one of two entangled Qbits, described in the Bell-states by the subscripts of A and B. If A decides to measure the related Qbits then the outcome cannot be predicted, as

$$\left|\frac{1}{\sqrt{2}}\right|^2 = 0.5 \quad (7)$$

and therefore with equal probabilities either $|0\rangle$ or $|1\rangle$ would be measured. However, according to quantum entanglement, B would now measure the same. This is because the final state $|0\rangle$ can only be the result of A's Qbit being in the state $|00\rangle$.

Quantum Bits

The introduced QANN relies on double values, so theoretically an 8 Qbytes or 64 Qbits-architecture needs to be simulated for being able to perform the task in one processing step, which would be the optimum. However, this would be oversized, as according to physicist David Deutsch, quantum parallelism allows a quantum computer to work on a million computations at once, while an ordinary PC works on one. A 30-qubit quantum computer would equal the processing power of a conventional computer that could run at 10 teraflops, where today's typical desktop computers run at speeds measured in

gigaflops (Bonsor, 2000). The probability distribution of a Qbit either being $|0\rangle$ or $|1\rangle$ after the collapse of $|\psi\rangle$ caused by measurement is, as already indicated, determined by complex in Hilbert space but not by real numbers. Therefore, for simulating one Qbit two floating point numbers are required. Theoretically, for state machines, like auto-associative artificial neural networks, a number of two architecture-Qbits would suffice, as their calculation is not based on floating point numbers, but on signed 2bit-values (-1, 0, 1).

Anyway, as the SHOCID quantum ANN is a simulation, the optimal architecture, which is a 64-bit one, has been simulated. Moreover, the number of used Qbits independent of the problem to be solved. Let assume, the QANN from Fig. 1 serves as model, then two registers have to be created, one representing the inputs $x_1, \dots, x_n$ and one representing the outputs $f(x_1, \dots, x_n)$. Besides, the process of carrying out the function $f(x)$ would need lots of additional Qbits which play an important part in the entanglement of the input and output register a quantum computer makes use of.

Quantum Artificial Neural Network

The structure of a quantum feed forward artificial neural network does not differ from a normal one, but the concept of linear superposition is one of the differences that can be described graphically by Fig. 1:

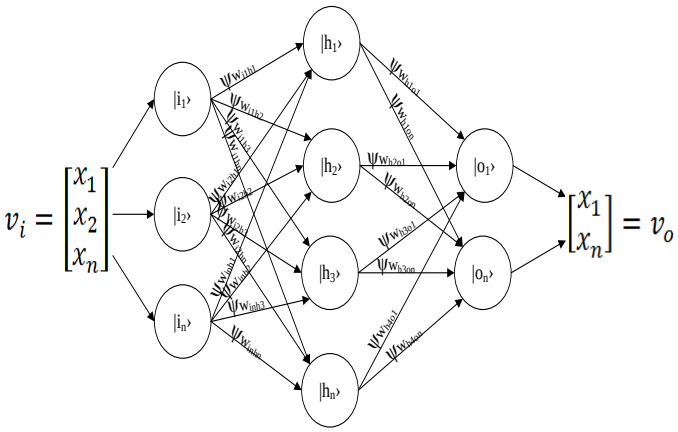

FIG. 1 – QUANTUM ARTIFICIAL NEURAL NETWORK

Fig. 1 shows that neurons have changed to quantum registers, which means that, referring to the picture above, the input registers $|i_1\rangle, \dots, |i_n\rangle$ hold any possible input dataset of the training data. Assuming a QANN shall be used to learn a problem based on input datasets consisting each of three attributes, it would also contain three input registers (input neurons in a standard ANN), each holding any value its related training data attribute. The same holds for the hidden layer, represented by $|h_1\rangle, \dots, |h_n\rangle$, whose quantum neuron registers hold any calculated, weighted sum of any possible input coming from $|i_1\rangle, \dots, |i_n\rangle$. Based on the training data the final layer $|o_1\rangle, \dots, |o_n\rangle$ holds any calculated output the neural network is capable of producing. Furthermore, the superpositions of the weight vectors are important to explain, as these hold every possible value they can represent, independent of the training data.

Based on the table Ezhov and Ventura provide in their work (Ezhov, Ventura) Tbls. 1 and 2 are of use for the understanding of the differences between ANNs on a classical computer and ANNs on a quantum computer.

TABLE 1 STANDARD ANN FEATURES

Feature	**Standard artificial neural network**	
Neuron encoding	Definite bit state	$x_i \in \{0,1\}$
Neuron connections	Weighted connections	$\{\omega_{ij}\}_{ij}^{n-1}$
Learning	Rule	e.g. $\Delta w_{ij}(t+1) = \mu\delta_j x_i + \alpha\Delta w_{ij}(t)$
Desired solution determination	Cost function	e.g. $n = \min\left(\sqrt[2]{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right)$
Result	$f(x)$	n

TABLE 2 QUANTUM ANN FEATURES

Feature	**Quantum artificial neural network**	
Neuron encoding	Probable Qbit state	$\|x\rangle = \alpha_1\|0\rangle + \alpha_2\|1\rangle$
Neuron connections	Weighted connections Entanglement	$\{\psi\omega_{ij}\}_{ij}^{n-1}$ $\|x_0 x_1 \dots x_{n-1}\rangle$
Learning	Superposition of entangled Qbits	$\sum_{i=1}^{n} \alpha_1 \|x_0^i \dots x_{n-1}^i\rangle$
Desired solution determination	Grover's search for optimal solution in superposition via unitary trans-formation	$U: \psi \to \psi'$ with an oracle like e.g. $\|p\rangle \geq n * m * p$
Result	Decoherence	$\sum_{i=1} \alpha_1 \|x^i\rangle \Rightarrow \|x^k\rangle$

Superposition with Respect to QANNs

A quantum artificial neural network does not only require to put its weights into superposition, so do its neurons, respectively their dynamic action potentials or thresholds as well as its underlying mathematical function.

Superposition of Dendrites

The dendrites or weights of a standard feed forward artificial neural network are lifted into linear superposition, which is the first step towards a quantum ANN. This means that every weight is changed into a wave and would theoretically contain every possible value at once. However, as currently the quantum ANN is simulated on a von Neumann machine, infinite accuracy, or better, any possible α_1 and α_2 in

$$|\psi\rangle = \alpha_1|0\rangle + \alpha_2|1\rangle = \begin{pmatrix}\alpha_1\\ \alpha_2\end{pmatrix} \quad (8)$$

restricted only by the normalization condition

$$|\alpha_1| + |\alpha_2| = 1 \quad (9)$$

would require exponentially more time than restricting the possible peculiarity of the weights by a range and increasing them only by pre-defined increments starting from the lower boundary to the upper boundary. Assuming the basis for the increment is 0.001 and the weight boundaries reach from -100 to 100, any value in this interval to the granularity of 0.001 will serve as possible weight. The process of setting the weights $w_0, \dots, w_n$ into $|\psi\rangle_0, \dots, |\psi\rangle_n$ with the exemplified parameters above happens as described in Tbl. 3.

TABLE 3 LIFTING WEIGHTS INTO SUPERPOSITION

Start

1. Set -100 as p_{start} and 100 as p_{end}
2. ***For each w***
   a) Set $w_{current}$ to p_{start}
   b) Repeat
      i. Save $w_{current}$ into corresponding $\psi n_f n_t$
      ii. Set $w_{current} = w_{current} + 0.001$
   c) Until p_{end} is reached

End

Breakdown:

$\psi n_f n_t$: Array list holding each possible normalized weight, being the superposition of a specific weight and n_f being the from-neuron, n_t being the to-neuron

p_{start} : Calculation start point

p_{end} : Calculation end point

$w_{current}$: The current weight, not normalized

Superposition of Neurons

The registers of each layer are lifted into linear superposition as well. Let say, an example QANN consists of three layers, like the one described with Fig. 1, and let further assume that any of the QANNs neurons scales its output through an activation function and makes use of dynamic thresholds, in which a learning algorithm would scale in standard ANN learning, then, according to the increment and upper and lower threshold boundaries, logically several thresholds for each superposition of networks must exist. Let further assume, the lower threshold boundary would be 0.1, the upper threshold boundary 1 and the increment 0.1, then 9 different values for a threshold would be possible. In the example QANN this leads to 387,420,489 (9^9) possible configurations of thresholds for one configuration of weights. However, as a specific configuration of weights only exists once within $|\psi\rangle$, the whole QANN must be lifted into superposition.

Overall Network Superposition

As the linear superposition indeed dictates that any possible configuration of each Qbit exists at once and as, related to the example QANN, any of these (weight) configurations has numerous configurations of thresholds, an impracticality occurs, namely the coexistence of identical configurations in one superposition. Thus, one configuration of a quantum artificial neural network must include both the weights and the neuron thresholds.

Processing

The calculations within the QANN need to be carried out as it is done in a standard FFANN. However, as indicated more than once, one of the major advantages of quantum mechanics is its superposition and the resulting quantum parallelism. For the QANN-internal calculations done by the function f on all configurations of x the reversible unitary transformation U_f, only taking basis states ($|0\rangle$ and $|1\rangle$) into such, will serve as example processing of one of the QANN's Qbits:

$$U_f(|x\rangle_n|y\rangle_m) = |x\rangle_n|y \oplus f(x)\rangle_m \quad (10)$$

where $\oplus$ indicates a modulo-2 bitwise addition or XOR. If x and y are m-bit integers whose j^{th} bits are x_j and y_j, then $x \oplus y$ is the m-bit integer whose j^{th} bit is $x_j \oplus y_j$. Thus,

$$1101 \oplus 0111 = 1010 \quad (11)$$

Furthermore, if the initial value represented by the output register is

$$y = 0 \tag{12}$$

then

$$U_f(|x\rangle_n|0\rangle_m) = |x\rangle_n|f(x)\rangle_m \tag{13}$$

and $f(x)$ would represent the result in the output register, where regardless to the initial state of y, the input register remains in its initial state $|x\rangle_n$. Furthermore, U_f fulfils another important criterion, which is that it is invertible:

$$U_f U_f|x\rangle|y\rangle = U_f(|x\rangle|y \oplus f(x)\rangle) = |x\rangle|y \oplus f(x) \oplus f(x)\rangle = |x\rangle|y\rangle \tag{14}$$

as

$$z \oplus z = 0 \tag{15}$$

for any z. Equation 4 shows an important feature of quantum computation, namely that the application of the 1-Qbit Hadamard transformation H on each Qbit in the 2-Qbit state results in

$$(H \otimes H)(|0\rangle|0\rangle) = H_1 H_0|0\rangle|0\rangle = (H|0\rangle)(H|0\rangle) = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) = \frac{1}{\sqrt{2}}(|0\rangle|0\rangle + |0\rangle|1\rangle + |1\rangle|0\rangle + |1\rangle|1\rangle) \tag{16}$$

which leads to the generalization of the n-fold tensor product of n Hadamard transformations on the n-Qbit state

$$(H^{\otimes n})(|0\rangle_n) = \frac{1}{2^{n/2}} \tag{17}$$

of first use in Eq. 19. If the initial state of an input register is $|0\rangle$, the application of n-fold Hadamard transformations transforms the state of this register into an equally weighted superposition of all n-Qbit inputs (Mermin, 2007).

Entanglement with Respect to QANNs

A possible implementation of a quantum artificial neural network could begin with a superposition of all possible weight vectors, which allows classifying all training examples with respect to every weight vector at once. Furthermore, a performance register $|p\rangle$ is used to store the number of correctly classified training examples and updated continuously. The update of the performance register with respect to each configuration of the QANN creates an entanglement of $|p\rangle$ and $|\psi\rangle$. Thus the oracle is

$$|p\rangle = n * m \tag{18}$$

where n represents the number of training examples and m the number of output neurons. As it may occur that either no configuration of a network within the superposition is able to classify all training examples correctly (and therefore every vector has an equal chance of being measured) or the amount of time required for finding a vector is increasing with the number of bits in the weight vector and thus exponential complexity:

$$O\left(\sqrt{2^b/t}\right) \tag{19}$$

For avoiding the first case, Ventura and Ricks suggest modifying the search oracle to

$$|p\rangle \geq n * m * p \tag{20}$$

where p is the percentage of correctly classified training examples.

With respect to quantum artificial neural networks, this means that any possible configuration of the quantum ANN is kept within the superposition. However, there is still the problem of measurement, as measurement needs to be done when probability of receiving a desired result is high. Let assume a quantum register consisting of 64 Qbits each in the already known state

$$|\psi\rangle = \frac{1}{2}(|0\rangle + |1\rangle) \tag{21}$$

then every possible state, or every value (double) that can be expressed with these 64 bits may be measured, but with the same probability distribution, so any of these double values would exist in this quantum register at once. Thus, an operator U_f, applying the already known ANN-internal calculations represented by the function f needs to be applied on $|\psi\rangle$, which writes the ANN results for being able to measure ANN-critical (performance)

$$x_{rmse} = \sqrt[2]{\frac{1}{n}\sum_{i=1}^{n} x_i^2} \tag{22}$$

resulting from $U_f|\psi\rangle$, into a second quantum register $|\phi\rangle$. $|\phi\rangle$ would then contain every possible result calculated from every configuration in $|\psi\rangle$. Furthermore, due to the calculation, the input and output registers will become entangled by the Qbits used by U_f, which in fact is a problem, as in this case both registers cannot be assigned a state on their own (see chapter 'Information about $f(x)$').

Information about $f(x)$

The nontrivial question is now, although both x and $f(x)$ are available, how to measure the required output, or in other words, how to know the content of a closed box without being allowed to look inside. As input and output are entangled now, a measurement of the output register would also let the input register

collapse and furthermore, no information about f can be gathered, which in fact represents the QANN. Thus, one has to dismiss the general opinion which does not require one to know what happens between the input and output neurons exactly, but to extract as many information about the internal calculations on all configurations of the QANN in $|\psi\rangle$ as possible. This may be achieved with unitary quantum gates applied on all registers before and after the application of U_f and by combining these with measurements of some values, thus subsets of Qbits, of the QANN. It is not a hundred percent clear now how this may happen, but when the whole QANN is measured, this would on the one hand theoretically allow gaining useful relations between x and f, and all this from one calculation, but on the other hand according to Heisenberg's uncertainty principle not allow determining $f(x)$ for a specific x. Shor, for example, has developed a quantum algorithm (which is of course probabilistic) for factorization and by the use of the discrete Fourier transformation. But back to QANNs, Mermin (Mermin, 2007), when explaining Deutsch's problem (Deutsch, 1985), gives a quite well example of how to reduce the probability of measuring a configuration, or better, gathering information of configurations of $f(x)$ that are not desired: let assume x specifies a choice of two different inputs to an elaborate subroutine that requires many additional Qbits, then one can think of f(x) as characterizing a two-valued property of this subroutine's output. It has also been mentioned that the input and output registers are not allowed to be entangled with the subroutine's Qbits after having finished processing, as in case of entanglement the input and output registers would not have final states on their own, which does not allow describing the computational process as unitary transformation at all. A simple linear transformation Eq. 4 may then be used to determine the net effects on the input and output register. The output

$$U_f(H\otimes 1)(|0\rangle|0\rangle) = \frac{1}{\sqrt{2}}|0\rangle|f(0)\rangle + \frac{1}{\sqrt{2}}|1\rangle|f(1)\rangle \quad (23)$$

was achieved by applying U_f to the input. Initially, the input and output registers need to be in the state $|0\rangle$, followed by the application of X on the input and output registers, again followed by the application of H on both. The input for U_f becomes then

$$(H\otimes H)(X\otimes X)(|0\rangle\otimes|0\rangle) = (H\otimes H)(|1\rangle\otimes|1\rangle) = \left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right)\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right) = \frac{1}{2}(|0\rangle|0\rangle - |1\rangle|0\rangle - |0\rangle|1\rangle + |1\rangle|1\rangle) \quad (24)$$

and the result

$$\frac{1}{2}U_f\big(U_f(|0\rangle|0\rangle) - U_f(|1\rangle|0\rangle) - U_f(|0\rangle|1\rangle) + U_f(|1\rangle|1\rangle)\big) \quad (25)$$

which in terms of the function $f(x)$ is then

$$\tfrac{1}{2}\big(|0\rangle|f(0)\rangle - |1\rangle|f(1)\rangle - |0\rangle|\tilde{f}(0)\rangle + |1\rangle|\tilde{f}(1)\rangle\big) \quad (26)$$

where

$$\tilde{x} = 1\oplus x \quad (27)$$

and

$$\tilde{1} = 0 \quad (28)$$

and

$$\tilde{0} = 1 \quad (29)$$

and

$$\tilde{f}(x) = 1\oplus f(x) \quad (30)$$

If

$$f(0) = f(1) \quad (31)$$

then the output state is

$$\tfrac{1}{2}\big(|0\rangle|f(0)\rangle - |1\rangle|f(1)\rangle - |0\rangle|\tilde{f}(0)\rangle + |1\rangle|\tilde{f}(1)\rangle\big) \quad (32)$$

or

$$\tfrac{1}{2}\Big((|0\rangle - |1\rangle)\big(|f(0)\rangle - |\tilde{f}(0)\rangle\big)\Big) \quad (33)$$

else if

$$f(0) \neq f(1) \quad (34)$$

then

$$\tfrac{1}{2}\big(|0\rangle|f(0)\rangle + |1\rangle|f(1)\rangle - |0\rangle|\tilde{f}(0)\rangle + |1\rangle|\tilde{f}(1)\rangle\big) \quad (35)$$

or

$$\tfrac{1}{2}\Big((|0\rangle + |1\rangle)\big(|f(0)\rangle - |\tilde{f}(0)\rangle\big)\Big) \quad (36)$$

H on the input register leads to

$$f(0) = f(1) \rightarrow |1\rangle\tfrac{1}{\sqrt{2}}|f(0)\rangle - |\tilde{f}(0)\rangle \quad (37)$$

and

$$f(0) \neq f(1) \rightarrow |0\rangle\tfrac{1}{\sqrt{2}}|f(0)\rangle - |\tilde{f}(0)\rangle \quad (38)$$

summed up as

$$(H\otimes 1)U_f(H\otimes H)(X\otimes X)(|0\rangle\otimes|0\rangle) = \begin{cases} |1\rangle\frac{1}{\sqrt{2}}|f(0)\rangle - |\tilde{f}(0)\rangle & , f(0) = f(1) \\ |0\rangle\frac{1}{\sqrt{2}}|f(0)\rangle - |\tilde{f}(0)\rangle & , f(0) \neq f(1) \end{cases} \quad (39)$$

This states that the input register is either $|0\rangle$ or $|1\rangle$,

but depends on what is true, Eq. 28 or Eq. 31. The clue is that two of four possible representations of the function f have been eliminated, just by one operation, which again is a quantum computer-specific ability. In terms of the QANN the same must now happen in more complex ways, as not only one Qbit needs to be taken into consideration and the internal calculations of an ANN require more than a few quantum operators and transformations. Currently, exactly this is subject of further research.

However, this is not enough. The transformation U_f does not consider the additional Qbits required for the calculations within the QANN. Thus, let assume a further transformation W_f, again inspired by Mermin's naming convention, which takes these bits into consideration. Whereas U_f only considered the Qbits of the input and output registers, W_f takes into account all Qbits. In terms of the QANN, this can be described perfectly well with Fig. 2:

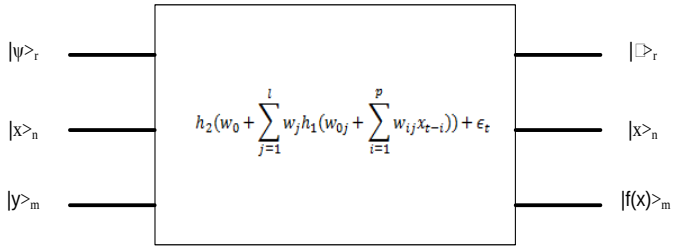

FIG. 2 – QUANTUM ARTIFICIAL NEURAL NETWORK CALCULATIONS

where the thick lines stand for multiple Qbit-inputs, with multiple Qbit-registers, as the calculation must be reversible. m and n represent the Qbits of the input and output registers, where r stands for the additional Qbits required for the QANN-internal computation. The r Qbits required for the computations are as well in superposition and after computation entangled with the input and output register. Thus the following quantum perceptron equations have to be set up:

Quantum single layer perceptron equation:

$$|f(x)\rangle = h(\sum_{i=1}^{n} |\omega_i\, x_i\rangle + \theta) + \epsilon_t \qquad (40)$$

where h is the activation function, applied to the summed weights in linear superposition $|\omega\rangle_i$ multiplied by the input values in linear superposition $|x\rangle_i$, if the threshold is exceeded. n represents the number of inputs. The activation function needs to be in superposition as well, as the dynamic action potentials of the neurons lead to multiple configurations of f. ϵ_t is the uncertainty variable of the ANN and additionally, the equation takes into consideration a bias θ, which is not necessarily used. This adapts Fig. 2 to Fig. 3:

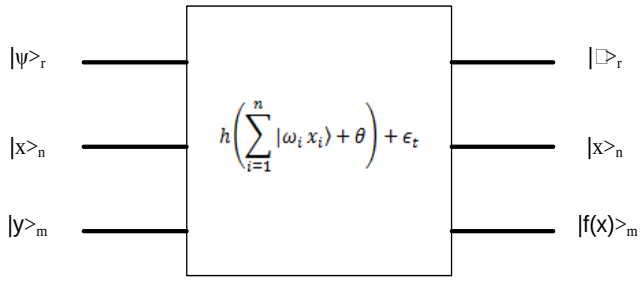

FIG. 3 – QUANTUM SINGLE LAYER PERCEPTRON DIAGRAM

Quantum multi layer perceptron equation:

$$|f(x_t)\rangle = h_2(w_0 + \sum_{j=1}^{l} |w_j\rangle h_1(w_{0j} + \sum_{i=1}^{p} |w_{ij}\, x_{t-i}\rangle)) + \epsilon_t \qquad (41)$$

where the input layer has p inputs $x_{t-1}, \ldots, x_{t-p}$, the hidden layer has l hidden nodes and the output, and there is a single output for the output layer x_t. Layers are fully connected by weights, where $|w\rangle_{ij}$ in linear superposition represents ith input for the jth node in the hidden layer, whereas $|w\rangle_j$ is the weight assigned to the jth node in the hidden layer for the output. w_0 and w_{0j} are the biases, while h_1 and h_2 are activation functions. Again, the dynamic action potentials of the neurons must be in linear superposition as well, which then leads to multiple configurations of not only the weights, but also the neurons. This finally leads to Fig. 4.

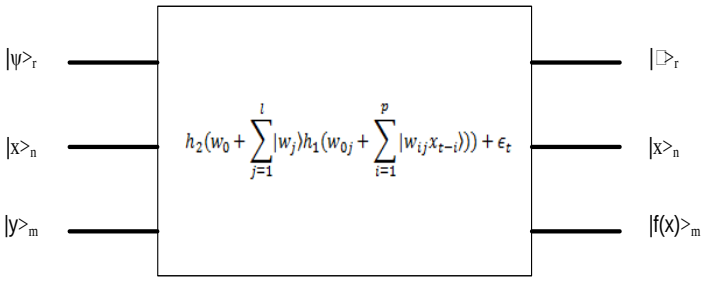

FIG. 4 – QUANTUM MULTI LAYER PERCEPTRON DIAGRAM

However, Fig. 4 may not be the final state, as this requires unentangled states of the registers. Moreover, in the initial state of the system, it is required to ensure the independency between the input and output registers' initial states and the additional calculation-Qbits initial states. Mermin suggests to achieve the independence of the calculation-Qbits by setting them to an initial state $|\psi\rangle_r$ and also taking care of the final state $|\phi\rangle_r$, which should be identical to $|\psi\rangle_r$, which can be achieved by reversible unitary transformations:

$$W_f = V_f^* C_m V_f \qquad (42)$$

or described graphically with Fig. 5:

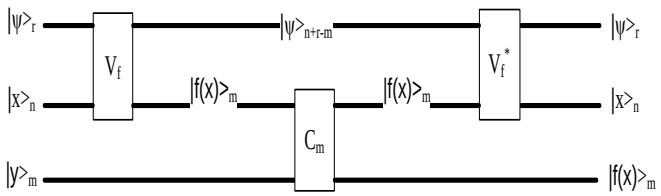

FIG. 5 – REVERSE THE CALCULATION BITS (MERMIN, 2007)

where V_f is a unitary transformation on the input register m and the calculation Qbit-register r, which creates an own state for both. Furthermore, this allows constructing the function $f(x)$ in a subset of the output register n, based on the entangled input and calculation Qbits. Then $|y\rangle_m$ is transformed into $|y \oplus f(x)\rangle_m$ without influencing the bits of the input register or the bits of the calculation register by the unitary transformation C_m. C_m does not affect the input or the calculation register, the inverse of V_f, namely V_f^* can be used to restore the initial state of $|\psi\rangle_r$ (Mermin, 2007). However, it is not always required to go that far, as the QANN is a mathematical function, where the variables in Fig. 2 are replaced by real numbers. Therefore, just for gathering information about f, it is not required to revert the unitary transformation V_f with V_f^*, but it is indeed required when there is the need to verify if f(x) equals f(y), which may be the case for an auto-associative QANN.

However, although the elimination of possible configurations of f decreases the probability of measuring an unwanted configuration and the time needed for quantum search, it does not reveal the desired configuration. Thus, a further register is required for measuring the performance of the quantum artificial neural network, for which continuous entanglement with the calculation register is absolutely necessary. After having applied the unitary transformations clearing away the entanglement between the input, calculation and output registers, there should only remain entanglement of the calculation and performance registers. When applying a quantum search on the performance register, which in fact manipulates the phases of the possible performance values until the probability of measuring the desired output is near unity, the following measurement also lets the calculation register collapse, which then reveals the desired function f, thus the quantum artificial neural network one is searching for (detailed explanation follows in the next chapter).

Measurement

Finally, all of this fails to consider the basic search problem, as a configuration of the QANN within the allowed parameters has to be found within its superposition. This is where Grover's algorithm (Grover, 1996) may be applied, which is used for searching a special configuration or item in an unsorted database (which is in case of the QANN the performance register in linear superposition). Grover's algorithm provides the answer to when the system shall be measured, as measurement lets the superposition collapse, which eliminates all possible configurations of the QANN except the measured one. It is important to mention that the algorithm is capable of finding the desired solution in $O(\sqrt{N})$ time (iterations), which is nothing that can be done on a von Neumann computer.

The algorithm will not be explained in detail here as this would go beyond the scope of this elaboration, but generally it does the following: it inverts the phase of the desired basis states followed by an inversion of all basis states about the average amplitude of all states. The repetition of this process produces an increase of the amplitude of the desired basis state to near unity, followed by a corresponding decrease in the amplitude of the desired state back to its original magnitude (Ezhov, Ventura). Grover detected that this routine just needs to be called a number of repetitions that does not exceed $\frac{\pi}{4}\sqrt{N}$, which is $O(\sqrt{N})$ iterations and thus, although the search is stochastic somehow, it outperforms a classical computer.

Although the algorithm is generally described as database search algorithm, it would be more suitable to describe it as function inverter. This is, because for a given function f(x) the algorithm is able to determine y.

Ricks and Ventura (Ricks, Ventura, 2003) made a very interesting proposal of how the optimal solution may be determined by a generalization of the of Grover's algorithm made of Boyer et al. (Boyer, 1996).

Another approach could be as follows: the output of a node would be $f(x)$ and according to Grover's algorithm the determination of y is possible, which has to happen with a quantum search routine U_f. This search routine must then calculate backwards through the network, which is quite different from any other approach in neural network learning. Usually, y is given and one tries to determine $f(x)$ and adapts $f(x)$ through a learning algorithm as long as it is required

to fulfil a stopping criterions, like the RMSE. However, as only $f(x)$ is given, U_f is required to find the correct input to the desired output. Thus, the calculated output must be taken and the calculation must go backwards. Let assume, the perceptron equation is as follows:

$$|f(x)\rangle = \tan\left(\sum_{i=1}^{n} |\omega\rangle_i |x\rangle_i + \theta\right) + \epsilon_t \quad (43)$$

Then U_f must be

$$|x\rangle_i = \frac{\arctan\left(|y\rangle_i - \epsilon_t\right) - \theta}{\sum_{i=1}^{n} |\omega\rangle_i} \quad (44)$$

and the error calculation

$$|y_{rmse}\rangle = \sqrt[2]{\frac{1}{n}\sum_{i=1}^{n} |y^2\rangle_i} \quad (45)$$

where n represents the number of outputs and y the input values.

However, again and before that, the input register consisting of a number of n Qbits has to be put into superposition as it has already been done in Eq. 11:

$$|\phi\rangle = H^{\otimes n}|0\rangle_n = \frac{1}{2^{n/2}}\sum_{x=0}^{2^n-1}|x\rangle_n \quad (46)$$

Also, the already introduced unitary transformation V is required, plus an additional unitary transformation W acting on the input register as V with a fixed form not depending on a and preserving the component of any state along the standard (initial) state $|\phi\rangle$, but changing the sign of its component orthogonal to $|\phi\rangle$:

$$W = 2|\phi\rangle\langle\phi| - 1 \quad (47)$$

$|\phi\rangle\langle\phi|$ representing the projection operator on $|\phi\rangle$ (Mermin, 2007). Given these to transformations, Grover's algorithm applies the Product WV many times onto the input register in $|\phi\rangle$. Furthermore, each invocation of WV requires the quantum search routine or unitary operator U_f to be executed, which must be able to work with the superpositions of the QANN's states and which compares the entries of the database, or in the case of a quantum artificial neural network, the desired output with the calculated output.

Summing up, in both algorithms the quantum search seeks to let the system fall into decoherence when the probability amplitudes for measuring the desired state near unity.

Envisaged Implementations of QANNs

There are several difficulties one faces when trying to implement a quantum computer. One of the major ones is the elimination of the physical system's (that a quantum computer certainly is) own irrelevant properties, so that the superposition, or coherence, can be established. This is not only difficult when thinking of possible implementations of a quantum artificial neural network, but a general issue. However, there may be a chance of implementing quantum systems capable of processing a quantum artificial neural network before standard quantum computers may be implemented. This is, because the first implementations will possibly target auto-associative artificial neural networks, not relying on double values for processing, but on but on signed 2bit-values values (-1, 0, 1). Thus, the architecture would not consist of 64 Qbits, but on 2 Qbits. Given such an auto-associative quantum artificial neural network for recognizing patterns consisting of 3 input values would just require 3 Qbits in case of a Hopfield artificial neural network (Fig. 6), and maybe 7 (depending from the number of the hidden classifiers) in case of a Boltzmann machine (Fig. 7).

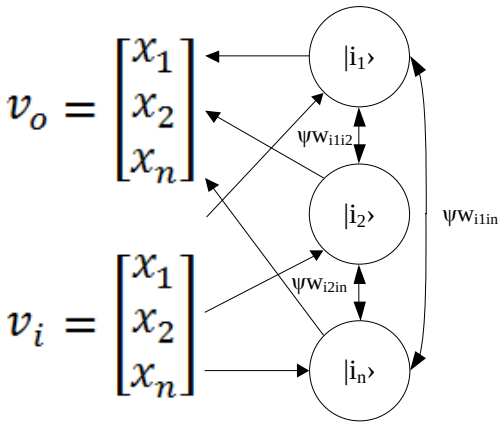

FIG. 6 – QUANTUM HOPFIELD ARTIFICIAL NEURAL NETWORK

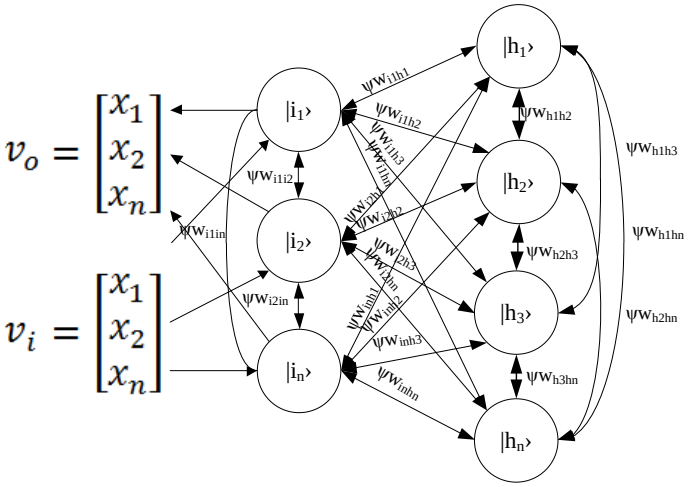

FIG. 7 – QUANTUM BOLTZMANN MACHINE

Another challenge is the realization of the required interconnection of the network's neurons. The

calculation of the connection weights happens in the perceptron equation and thus creating entanglement of the calculation bits. Therefore, the weighted neuron connections exist as entangled Qbit-states. According to DiVincenzo, physical systems implementing quantum computers must provide the following (DiVincenzo criteria):

- A scalable system of well-characterized Qbits,
- which must offer the possibility of proper initialization and
- which must have much longer coherence times than the time scales required for the fundamental operations,
- a universal set of quantum gates and
- a Qbit-specific measurement.

Additionally, DiVincenzo demands for

- the ability to interconvert stationary and flying Qbits and
- the ability to faithfully transmit flying Qbits between specified locations (DiVincenco, 2001).

Technically, there exist no fundamental obstacles in implementing a fault-tolerant quantum computer; however, there is no best approach, although a few top candidates exist. The implementation of a quantum system being capable of processing an artificial neural network may be realized by the use of the following systems:

Nuclear Magnetic Resonance

NMR seems to be a promising approach for implementing a quantum artificial neural network, as although the number of bits is currently limited, linear superposition can be kept up to thousands of seconds, thus long enough for allowing calculation, quantum search and measurement. Chuang and Gershenfeld (Gershenfeld, 1996) made an experimental verification using a nuclear magnetic resonance system made up of two Qbits, the first having been the spin of a carbon isotope's (C^{13}) nucleus, the second having been described by the spin of a hydrogen ion's (H^{+}) proton spin. However, although their system is a real quantum system, they compute the statistical average of many copies of the system, thus molecules.

Others

Other promising systems are quantum dot systems, consisting of an electron, trapped in an atomic cage, ion traps, or quantum electrodynamics of atoms in optical cavities.

Results

The simulation of the quantum artificial neural network on a von Neumann computer has been successfully realized, however, the thoughtful reader will understand that no results except this can be provided here. After having lifted the artificial neural network into a restricted superposition (see Tbl. 3), the time-consuming quantum database search has been carried out, which in fact proofed that the desired configuration could be found in any case according to Eq. 42. However, unless the theorem can be implemented on a quantum computer it remains a theorem waiting for a final proof.

Conclusion

Quantum computer science offers some very interesting effects in terms of processing information, like entanglement, which does not have a counterpart in classical computation. In terms of quantum artificial neural networks, the difficulty is to measure the system that is to destroy the superposition, exactly when the system's state fulfils the required learning criteria, e.g. the number of correctly classified training examples. Moreover, when taking the internal calculations, dynamic thresholds and the weights into consideration, the possible configurations of the ANN within its superposition grow exponentially in the size of the combined input vector plus threshold values.

However, quantum computer science does not only offer new approaches to information processing; it opens the door to the future of artificial intelligence. The massive parallel processing quantum computers are capable of resembling some kind of the parallel information processing of the human brain. Thitherto, it is not fully understood how information in the human brain is processed, not to mention the emergence of human consciousness. The brain is such a complex structure that not only its continuous study will lead to the full understanding, but also several experiments. These experiments may in the near future not only consist of invasive, semi-invasive or non-invasive approaches, but also in the simulation of a brain with all of its ~100,000,000,000 neurons, but also its capability to process information parallel with quasi one processor. Although the human brain that may process 2^{13} analog operations, is in terms of these operations already slower than current supercomputers, it does not require that much energy (only 15-20 watts). Furthermore, the whole neural network allowing unique information processing has

more than once been declared to be the most complex information system and thus it is a structure worth being imitated. It is very likely that this will be possible and happen sometime with a quantum computer simulating a biological neural network by a corresponding artificial one.

ACKNOWLEDGMENT

Thanks to Professor Dr. Ing. Sorin-Aurel Moraru for supporting the SHOCID research and development project.

REFERENCES

Bell, J. S. (1987): Speakable and Unspeakable in Quantum Mechanics, Cambridge: Cambridge University Press

Bonsor, K., Strickland, J. (2000): How Quantum Computers Work [2012-10-18], URL: http://computer.howstuffworks.com/quantum-computer.htm

Boyer, M. et al. (1996): Tight Bounds on Quantum Searching, Fourth Workshop on Physics and Computation.

Deutsch, D. (1985): The Church-Turing principle and the universal quantum computer; Proceedings of the Royal Society of London A. 400, 1985, p. 97.

Ezhov, A., Ventura, D. (-): Quantum neural networks, BSTU Laboratory of Artificial Neural Networks

Grover, L. K. (1996): A fast quantum mechanical algorithm for database search, Proceedings of the 28th Annual ACM Symposium on the Theory of Computation, pp.212-219.

Heaton Research: The number of Hidden Layers, [2012-13-01]; URL: http://www.heatonresearch.com/node/707

Mermin, D. N. (2007): Quantum Computer Science: An Introduction; Cambridge: Cambridge University Press

Neil, Gershenfeld, L., Chuang Isaac (1996): Quantum Computing with Molecules [2012-12-19]; URL: http://www.mat.ucm.es/catedramdeguzman/old/01historias/haciaelfuturo/Burgos090900/quantumcomputingSciAmer/0698gershenfeld.html

Neukart, F. et al. (2010): High Order Computational Intelligence in Data Mining - A generic approach to systemic intelligent Data Mining, Proceedings of Speech Technology and Human-Computer Dialogue (SpeD), 2011 6th Conference on, 2011, pp. 1-9.

P., DiVincenzo David. (2001): Dogma and heresy in quantum computing; Quantum Information Comp. 1, 1.

Ricks, B., Ventura, D. (2003): Training a Quantum Neural Network; Provo: Brigham Young University.